User Semantic Model for Hybrid Recommender Systems

Sonia Ben Ticha
URPAH & KIWI-LORIA Teams
Tunis, Tunisia & Nancy, France
e-mail: sonia.benticha@loria.fr

Azim Roussanaly
KIWI Team
LORIA laboratory
Nancy, France
e-mail: azim.roussanaly@loria.fr

Anne Boyer
KIWI Team
LORIA laboratory
Nancy, France
e-mail: anne.boyer@loria.fr

***Abstract*— Recommender systems provide relevant items to users from a large number of choices. In this work, we are interested in personalized recommender systems where user model is based on an analysis of usage. Collaborative filtering and content-based filtering are the most widely used techniques in personalized recommender systems. Each technique has its drawbacks, so hybrid solutions, combining the two techniques, have emerged to overcome their disadvantages and benefit from their strengths. In this paper, we propose a hybrid solution combining collaborative filtering and content-based filtering. With this aim, we have defined a new user model, called user semantic model, to model user semantic preferences based on items' features and user ratings. The user semantic model is built from the user-item model by using a fuzzy clustering algorithm: the Fuzzy C Mean (FCM) algorithm. Then, we used the user semantic model in a user-based collaborative filtering algorithm to calculate the similarity between users. Applying our approach to the MoviesLens dataset, significant improvements can be noticed comparatively to standards user-based and item-based collaborative filtering algorithms.**

Keywords-Collaborative filtering; semantic attribute; fuzzy clustering; hybrid recommender system.

I. INTRODUCTION

Recommender Systems (RS) provide relevant items to users from a large number of choices. Several recommendations techniques exist in the literature. Among these techniques, there are those that provide personalized recommendations by defining a profile for each user. In this work, we are interested in personalized recommender systems where the user model is based on an analysis of usage. This model is usually represented by a user-item ratings matrix, which is extremely sparse (> 90% of missing data).

Collaborative filtering (CF) [11] has been the first personalized recommender system. In CF, user will be recommended items that people with similar tastes and preferences liked in the past. Content-based filtering (CB) [5] is another important technique of recommendation; it assumes that each user operates independently. In content-based recommender systems, user will be recommended items similar to the ones he preferred in the past. CB uses techniques developed in information retrieval and information filtering research. The major difference between CF and CB recommender systems is that CF uses only the user-item ratings data to make predictions and recommendations, while content-based recommender systems rely on the features of items (semantic information) for predictions.

However, those techniques must face many challenges [13], like the *data sparsity* problem due to missing data in user-item matrix; the *scalability* problem for big database with great number of users and items; the *cold start* problem when new user logs in, the system ignores he or her preferences. Furthermore, each technique introduced its own shortcomings. In CF technique, if new item appears in the database, there is no way to be recommended before it is rated, this problem is known also as Cold-start problem. *Neighbor transitivity* refers to a problem with sparse databases, in which users with similar tastes may not be identified as such if they have any items rated in common. CB filtering suffers a problem of over-specialization where a user is restricted to seeing items similar to those already rated.

To overcome the disadvantages of both techniques and benefit from their strengths, hybrid solutions [4] have emerged. Most of these hybrid systems are process-oriented: they run CF on the results of CB or vice versa. CF exploits information from users' ratings. CB exploits information from items and their features. However, they miss the relation between users' ratings and items' features. This link may explain the user interests for an item.

In this paper, we propose a hybrid solution, adopting CF principle, by introducing in its recommendation process, data from usage analysis and semantic information of items. Our solution builds a new user model, *user-semantic model*, to model user preferences based on items content (semantic information). Therefore, our user model is the link between users' ratings and items' content by modeling users' features preferences.

The *user-semantic model* is built from users' ratings and items features by using machine learning: the Fuzzy C Mean (FCM) algorithm [2]. This model is used in a user-based CF algorithm to calculate the similarity between users. We have compared our results to the standards user-based CF [9] and item-based CF [10] algorithms. Our approach results in an overall improvement in prediction accuracy.

Our contribution is summarized as follows: (i) we construct a novel *user-semantic model*, representing the link between user's preferences and item's features, (ii) we use a machine learning algorithm, the Fuzzy C Mean clustering algorithm, FCM, for the construction of this model, (iii) we provide predictions and recommendations by using the user-semantic model, in a user-based CF algorithm [9], for

computing similarity between users, (iv) we perform several experiments with MoviesLens data sets, which showed improvement in the quality of predictions compared to user-based CF, item-based CF and a hybrid algorithm.

The rest of the paper is organized as follows: Section 2 summarizes the related work. FCM algorithm is described in Section 3. Standard user-based CF is described in Section 4. Section 5 describes our approach and experimental results are given in Section 6. Finally, we conclude with a summary of our findings and some directions for future work.

II. Related Work

Recommender systems have become an independent research area in the middle 1990s after the apparition of the first paper on personalized recommender systems based on collaborative filtering [20]. Collaborative filtering is the most widespread used technique in recommender systems. It was the subject of several researches [3][9][10][19].

Purely content-based recommender systems are less widespread. Techniques used are from information retrieval and information filtering research. Notable works can be found in [21][22][23].

Several recommender systems use a hybrid approach by combining collaborative and content-based methods, which helps to avoid certain shortcomings of CB and CF systems. The Fab System of Balabanovic [1] counts among the first hybrid RS. Many others systems have been developed since [14][15][16][17][18]. A comprehensive survey of hybrid recommender systems can be found in Burke [4]. Most of these hybrid systems are process-oriented: they run CF on the results of CB or vice versa. In [7], authors integrate semantic similarities of items with user-rating similarities. The combined similarity measure was used in an item-based CF to generate recommendations. These works ignore the dependency between users' ratings and items' features. Taking account of the link between them can improve the quality of recommendation. In [12], this dependency was computed using the *term frequency/inverse document frequency* (TF-IDF) measure that is the most widespread measures for specifying keyword weights in Information Retrieval. The authors use this measure to calculate the weight of feature for each user. For computing this weight, they use only the items liked by the user; which forces to define a rating value threshold to select the items preferred by user. This solution has two shortcomings; first, the threshold value is very subjective, rating value 3 for an item on a scale from 1 to 5, can be a good value for a user and average value for another. Second, two users share the same tastes when, not only, they like the same things, but also, when they hate the same things; but in this approach, items not liked by users are not selected.

III. Fuzzy C Mean Algorithm (FCM)

The FCM algorithm is one of the most widely used fuzzy clustering algorithms. This technique was originally introduced by Jim Bezdek in 1981 [2]. The FCM algorithm is similar to the k-mean algorithm, but it provides non-disjointed clusters. It attempts to partition a finite collection of M elements, defined in N dimensional space, $E=\{X_1, X_j, X_M\}$ into a collection of L fuzzy clusters with respect to some given criterion. Given a finite set of data, the algorithm returns:

- a list of L cluster centers C_k such that $C_k=c_{k,i}, i=1,\ldots,N$
- a partition matrix P such that: $P=P_{kj}$, $k=1,\ldots,L$ and $j=1,\ldots,M$, P_{kj} is a coefficient $\in [0,1]$ giving the degree to which the element X_j belongs to the k-th cluster. Usually, the sum of those coefficients for any given element X_j is defined to be 1 as shown in (1).

The center of cluster Ck is the mean of all elements X in E, weighted by their degree of belonging to the cluster k (2).

$$\forall\, X_j \in E, \left(\sum_{k=1}^{L} p_{k,X_j} = 1 \right). \tag{1}$$

$$C_k = \frac{\sum_{X_j \in E} p_{k,j}^{m} X_j}{\sum_{X_j \in E} p_{k,j}^{m}}. \tag{2}$$

The coefficient of belonging is related to the inverse of the distance to the cluster center. In (3) the coefficient is normalized and "fuzzyfied" with a real parameter $m>1$ so their sum is equal to 1.

$$p_{k,j} = \frac{1}{\sum_{i=1}^{L} \left(\frac{distance(C_k, X_j)}{distance(C_i, X_j)} \right)^{2/(m-1)}} \tag{3}$$

The FCM algorithm consists of the followings steps:

-Choose a number of clusters, L

-Assign to each element $X_{j,\ j=1..M}$ coefficients $P_{k,j}$ of belonging to cluster $k_{,\ k=1..L}$.

- Repeat until the algorithm has converged:
  * Compute the center for each cluster, using the formula given by (2).
  * For each element, compute its coefficients for being in clusters, using the formula given by (3).

IV. User based CF algorithm

In collaborative filtering, active user (indicated with a subscript a) will be recommended items that people with similar tastes and preferences liked in the past. Breese et al. [3] have identified two classes of CF algorithms: *memory-based* and *model-based* algorithms. Memory-based algorithms use the entire of the user-item matrix to generate predictions. This allows them to be very reactive, by integrating immediately modifications of users' profiles into the system. However, even if these methods work well with small-sized database, Breese et al. [3] think that their scalability is problematic for big databases with great number of items and/or users. The *model-based* algorithms constitute an alternative to this problem. These algorithms build descriptive models via a learning process. Then, predictions are inferred from these models.

User-based [9] and Item-based, introduced by Sarwar et al. in [10], algorithms are the most prevalent memory-based

methods. They are both based on the *k*-Nearest-Neighbors algorithm. The first computes similarities between users and in the second, similarities are computed between items. In our approach we have applied the user-based CF algorithm for recommendation. Therefore, in this section, we describe its principle that consists of the following steps:

- *Calculate the similarities sim(u_a,v):* which reflect the correlation between the active user u_a and all others users *v*. The similarity is computed by the Pearson correlation (4), introduced by Resnick et al. [9].
- *Compute the predictions pr(u_a,i):* predict the rating value of active user u_a on non rated item *i*. In the user-based *CF* algorithm, a subset of nearest neighbors of the active user u_a are chosen based on their similarity with him or her, and a weighted aggregate of their ratings is used to generate predictions for the active user. Formula for computing predictions is given in (5).
- *Recommendation*: the system recommends to the active user, items with predicted ratings greater than a given threshold.

$$sim(u_a, v) = \frac{\sum_i (r_{u_a,i} - r_{u_a})(r_{v,i} - r_v)}{\sqrt{\sum_i (r_{u_a,i} - r_{u_a})^2} \sqrt{\sum_i (r_{v,i} - r_v)^2}} \quad (4)$$

where the *i* summations are over the items that both users u_a and *v* have rated and, r_v is the average rating of the rated items of the user *v*.

$$pr(u_a, i) = r_{u_a} + k \sum_{v \in V} sim(u_a, v)(r_{v,i} - r_v) \quad (5)$$

$$where \quad k = \frac{1}{\sum_{v \in V} |sim(u_a, v)|}$$

V denotes the set of the most similar users to the active user u_a (called the nearest neighbors) that have rated item *i*. *V* can range anywhere from 1 to the number of all users.

V. PROPOSED APPROACH

Several personalized recommendations techniques have been developed throughout recent years. CF is one of the most prevalent and older technique. Although CF has been very successful, it must face several challenges, like the data sparsity, the scalability and the cold start problems. One of the prospected solutions is to develop hybrid recommendation strategies that combine CF with CB filtering techniques.

We propose a personalized hybrid RS, adopting the principle of collaborative filtering, which includes in its recommendation process data from usage analysis, the users' ratings, and semantics information from items.

Fig. 1 shows the architecture of our system. It consists of two components:

- *Building the user semantic model*: provides the user semantic preferences by taking into account the dependency between users' ratings and items' features. This model is represented by a users-features matrix that predicts for each user a rating for each feature. This matrix, has no missing value, and is used to calculate the similarities between users in the recommendation process. This processing will be computed offline.
- *Computing predictions and Recommendations:* we predict for each user a list of relevant items based on the user-based CF algorithm. Similarities between users are computed, by using the user-feature matrix instead of the user-item rating matrix.

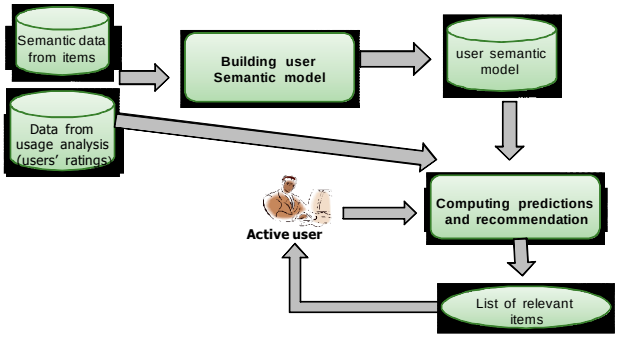

Figure 1. Architecture of our system

Although we apply a user-based CF for recommendation, our approach is also a model-based method. Indeed, user-semantic model is built from usage and semantic data, and used to predict ratings of active user on non rated items. This model allows a dimension reduction since the number of columns of the user-feature matrix is much lower than those of user-item rating matrix. Furthermore, the computing of similarities between users can be done offline. Thus, only the computing of predictions will be done online. For these reasons, our approach resolves the scalability problem.

Beside the scalability problem, our algorithm alleviates the data sparsity problem. Indeed, the user-features matrix, modeling the user semantic preferences, is a full matrix containing no missing value, thus, all similarities can be computed. This is not the case with user-item ratings matrix, because similarities between users who have no co rated items cannot be computed.

In the following sections, we describe each component in detail. All used symbols are described in Table I.

A. Building the users semantic model

Before defining the semantic profile of a user, we need to give some definitions. In this work, we suppose that items are described by a structured data in which there is a small number of attributes, each item is described by the same set of attributes, and there is a known set of values that each attribute may have. For example, the attributes of a movie can be: *title*, *genres*, *actors*, *director*. An attribute may have many values; each value is called *feature* or *semantic attribute* (used by Mobasher et al. in [7]). For a same item, if an attribute has many values (features), it is called *multi-valued attribute (genres* and *actors* in a *movie*), while if it must has only one value, it is called *mono-valued attribute* (*director* in a *movie*). The semantic profile of user *u* is then a vector A_u that each column, $a_{u,f}$ provides preference of *u* to a corresponding feature *f*. (value or semantic attribute) This preference is a real number like the rating value. In the

TABLE I
DESCRIPTION OF THE USED SYMBOLS

Symbol	Meaning	Description
N	Number of users	
M	Number of items	
L	Number of features	
U	The use-items ratings matrix	Missing values
$I=U^t$	The items-users ratings matrix	U transposed
U_u	The ratings vector of user u	The user's profile
I_i	The ratings vector of item i by all users	Item usage profile
F	The Items-features matrix	No missing value
F_i	The features vector of item i	Item semantic profile
$b_{i,f}$	The value of item-feature matrix	0 or 1
A	The users-features matrix	result of our approach
A_u	The user-features profile of user u	
?	Missing value	
$r_{u,i}$	Rating of user u on item i	
f	Feature	
i	Item	
u	User	
P_{ki}	The degree of item i of being in the cluster k	Fuzzy C Mean
nbMinR	Minimum Number of rating for items selected in computing the initial center of clusters	Fuzzy C Mean

following, we build *user-semantic model* for only one attribute and we suppose that it is multi-valued.

The usage analysis profile of user *u* is defined by a ratings vector: $U_u=(r_{u,1},r_{u,2},\ldots,r_{u,i},\ldots,r_{u,m})$.

The profile of item *i* is defined by 2 vectors, the first is based on usage analysis (ratings) and the second on features (values) of an attribute:

- Item usage based profile: a ratings vector: $I_i=(r_{1,I},r_{2,i},\ldots r_{u,i},\ldots,r_{N,i})$
- Item semantic based profile : a features vector for a multi-valued attribute: $F_i=(b_{i,1}, b_{i,2},\ldots,b_{i,L})$ where:

$$b_{i,f} = \begin{cases} 0 \text{ if item } i \text{ hasn't feature (value) } f \\ 1 \text{ if item } i \text{ has feature (value) } f \end{cases}$$

User-semantic model is defined by a *user-feature matrix* *A* (*N* lines and *L* columns) as shown in Fig. 2. Each line describes the semantic preferences of a corresponding user *u* and is defined by the vector $A_u=(a_{u,1},\ldots,a_{u,f},\ a_{u,L})$, *L* is the numbers of features and $a_{u,f}$ indicates the preference of user *u* on feature *f*.

The semantic profile of users is computed from items semantic profiles and users' ratings. For example, assume that we have a movies Data set with users ratings, and we want to predict the preference of user *u* for the *action movies* This means calculating an aggregation overall ratings of user *u* on all *action movies*:

$$a_{u,genre=action} = AGGR_{\ i.genre=action}\ r_{u,i} \quad (6)$$

The aggregation function can be a simple function like the average (*AVG*), or more complicated mathematical, or special user-defined function. In our approach, we choose to define a special user function, so we use a machine learning algorithm to learn the user semantic profile.

The idea is to partition items, defined by their usage profiles, in *L* clusters. Each cluster represents a feature (semantic attribute) of the selected attribute. Thus, the center of each cluster provides the profile of the corresponding feature in *N* dimensional space. In *movies* dataset, if we choose the attribute *genre*, then each cluster regroups movies with same *genre* (*action*, *children's*, *comedy*…) and its center provides the profile of the corresponding genre in *N* dimensional space.

However, as we have already said, the attribute is multi-valued, thus, the partition cannot be disjointed (for the *movie* data set, a same *movie* can have many *genres*; then it can belong to several clusters). For this case we need to use a fuzzy clustering algorithm to provide non-disjointed clusters. After a study of several fuzzy clustering algorithms we choose de Fuzzy C Mean (FCM) for its simplicity especially the number of clusters in our case is known and it's equal to the number of features.

Items semantic profiles are used for initializing the centers of clusters and the partition matrix.

The construction of user semantic model consists of 2 steps as shown in Fig. 2:

1. Clustering items, represented by their usage based profile, by using the FCM algorithm. This step provides L non-disjoints clusters represented by their center $C_k=(c_{k,1},c_{k,2},\ldots,c_{k,N})$ $k \in [1,L]$. The profile of feature k is then the vector C_k and *C* is a centers-clusters matrix, as

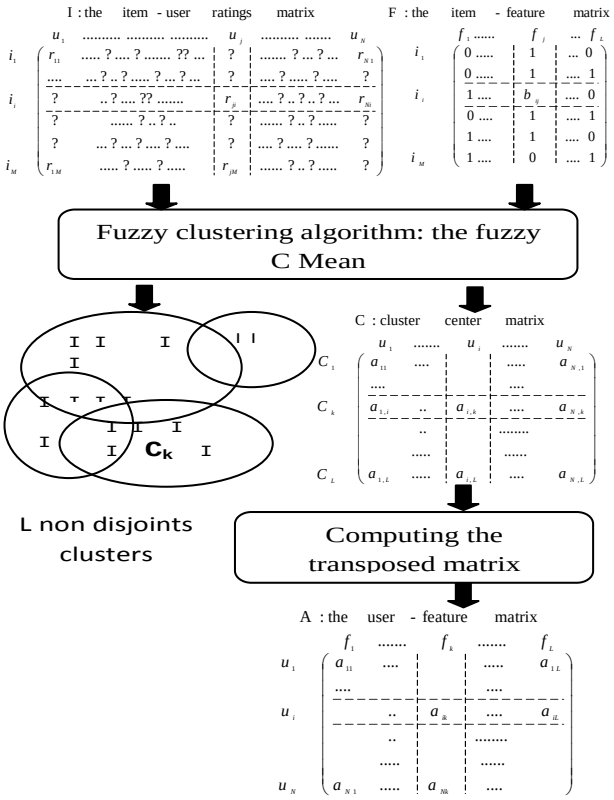

Figure 2. The fuzzy clustering algorithm provides L non-disjoint clusters. C_k is the centroid of the cluster k, C_k is profile of the feature f_k, defined in user dimensional space N. $C_k=A^t_k$

shown in Fig. 2, defined in *L (features)*x*N (users)* dimensional space.

2. Computing the transposed matrix of *C*: each line in *C* defined the corresponding feature profile. The transposed of *C* gives the matrix *A*, the user-feature matrix. *A* provides for each user *u* his or her features' preferences for the selected attribute. The matrix *A* has no missing value, what allows computing the similarity between all users. This is not the case when matrix has missing values.

In our approach, user semantic model is built by applying the collaborative principle. Indeed, in clustering process, items are defined by their usage based profiles. So, semantic profile of a user is computed from the ratings of all users, and not from his or her ratings only. This is not the case of the simple aggregation like the *AVG* function.

1)Initialization of the partition matrix of FCM algorithm

Semantic profiles of items are used to initialize the partition matrix *P*. In our algorithm, we initialize this matrix with respect to the formula given in (1). We use, for that, the items-features matrix *F*, then the degree to which the item i belongs to cluster k is given by (7)

$$p_{k,i}\begin{cases} = \dfrac{1}{\sum_{j=1}^{L} b_{i,f_j}}, \text{if} \quad b_{i,f_k} = 1 \\ = 0, \quad \text{if } b_{i,f_k} = 0 \end{cases} \qquad (7)$$

Example:

	f_1	f_2	f_3
i_1	0	1	1
i_2	1	0	1
i_3	1	1	1
i_4	1	1	0

In this example, we have three features, so 3 clusters. Item i_4 belongs to two clusters 1 and 2. Thus, $p_{3,4}=0$ because $b_{4,3}=0$, that means i_4 hasn't f_3; $p_{2,4}=0.5$ and $p_{1,4}=0.5$ because item i_4 has two features for the selected attribute. Thus, $p_{1,4}+p_{2,4}+p_{3,4}=1$. We assume that all features of a same item have the same weight. This assumption can be changed if we have information about the weight of each feature in item.

2)Initialization of the cluster centers

The initialization of the clusters centers is deducted from the partition matrix *P* by the formula given in (2). To compute the initial centers of clusters, we select only items having a number of ratings higher than the threshold *nbMinR* that its value will be defined empirically.

B. Computing predictions and Recommendation

To compute predictions for the active user, we use the user-based CF algorithm described in Section IV. In the standard user-based CF algorithm, the users-items matrix is used to compute users' similarities (see formula given in (4)). In our algorithm, we use the users-features matrix instead; thus, formula given in (8) is used to compute similarities. This allows inferring similarity between two users even when they have any co-rated items because the users-features matrix has no missing value. Thus, our approach provides solution to the neighbor transitivity problem emanates from

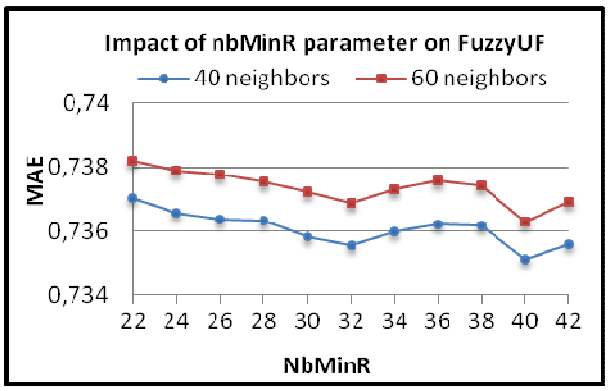

Figure 3. Impact, of the minimum number of ratings in initilization of clusters' centers on recommendation accuracy.

the sparse nature of the underlying data sets. In this problem, users with similar preferences may not be identifies as such if they haven't any items rated in common.

$$sim(u_a,v)=\frac{\sum_f (a_{u_a,f}-a_{u_a})(a_{v,f}-a_v)}{\sqrt{\sum_f (a_{u_a,f}-a_{u_a})^2}\sqrt{\sum_f (a_{v,f}-a_v)^2}} \qquad (8)$$

where the *f* summations are over features, and a_{Ua} is the average of $a_{ua,f}$,$f=1,\ldots,L$.

VI. Performance Study

In this section, we study the performance of our algorithm, called *FuzzyUF* on the figures, against the standard User-Based CF (*UB*), the standard Item-Based CF (*IB*) [10] and Average User Feature algorithm (*AvgUF*). For *IB* algorithm, we compute predictions using the *Adjusted Cosine* correlation measure which provides, according to [10], best prediction accuracy. The *AvgUF* is building user semantic profile by using the average (*AVG*) as an aggregation function (see formula in (6)). We evaluate these algorithms in terms of predictions relevancy.

A. The used corpus and experiments

In order to compute the prediction relevancy in our system, we used the data set of the MovieLens recommendation system [8], collected by the GroupLens Research Project [24]. This Data set contains 100,000 real ratings on 1682 movies from 943 users. Items are movies, and the used attribute is movie's genre that has 18 features (Action, Romance, Horror, etc.). Each user has rated at least 20 items. The data set has been divided into a training set (including 80% of all ratings) and a test set (20% of all

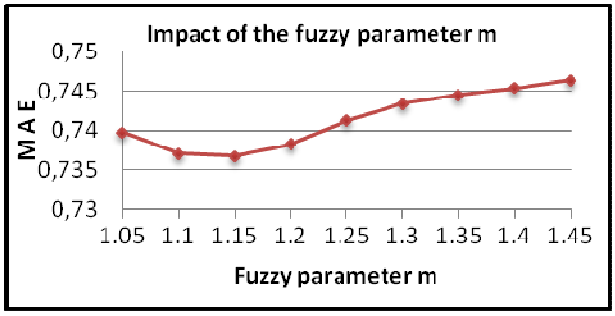

Figure 4. Impact of the parameter m on recommandation accuracy

ratings). We use the five training and test set (*u1 to u5*) provided by GroupLens for cross validation. Thus, we repeat the experiment with each training and test set and we average the results.

For the FCM algorithm, after having tried several distance measures, the Manhattan distance provides the best result. We have executed the FCM for many values of the fuzzy parameter *m*, and *nbMinR*. In all experiments, the FCM has converged to a global minimum.

B. Results

We evaluate our algorithm by using the *Mean Absolute Error* (MAE) [6]. MAE is the most widely used metric in CF research literature, which computes the average of the absolute difference between the predictions and true ratings, as shown in (9).

$$MAE = \frac{\sum_{u,i} \left| pr_{u,i} - r_{u,i} \right|}{d} \quad (9)$$

where d is the total number of ratings over all users, $pr_{u,i}$ is the predicted rating for user u on item i, and r_u, i is the actual rating. Lower the MAE is, better is the prediction.

In Fig. 3, the MAE has been plotted with respect to the minimum number of ratings for selecting items in initialization of the centers of the clusters (*nbMinR*). In both cases (40 and 60 neighbors), the MAE converges for 40 ratings. This plot shows the impact of *nbMinR* on the accuracy of the recommendations, which is expected since the number of item ratings influences the accuracy of the user semantic profile. Fig. 4 shows that small values of *m* improve the accuracy of our algorithm. The reason is when *m* is close to 1, then the cluster center closest to the items is given much more weight than the others and the FCM is similar to K-means algorithm. The minimum is obtained for *m*=1.15.

Fig. 5 depicts the prediction accuracy of our algorithm in contrast to those produced by *IB*, *UB* and *AvgUF*. In Fig. 5, the MAE has been plotted with respect to the number of neighbors in the *k*-Nearest-Neighbors algorithm. In all cases, the MAE converges between 30 and 40 neighbors, however, our algorithm results in an overall improvement in accuracy.

The improvement of accuracy can be explained by many reasons. First, taking into account the semantic profile of items in the recommendation process. Second, user semantic model is built according to a collaborative principle; ratings of all users are used to compute the semantic profile of each user. It is not the case of the *AvgFU* algorithm; this may explain its results despite taking into account the semantic aspect. Third, the choice of the attribute can have significant influence on improving the accuracy. Indeed, movie genre represents a very important evaluation criterion for users. Lastly, users-features matrix has no missing value, so, it allows inferring similarity between two given users even when they have any items rated in common.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a hybrid solution combining collaborative filtering and content-based techniques. The contribution of our solution over the

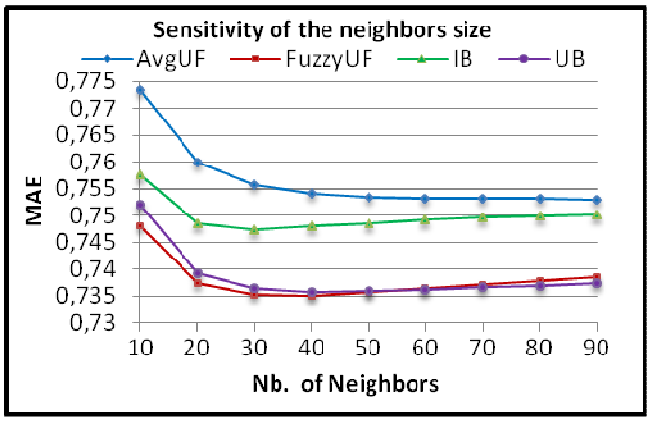

Figure 5. Prediction accuracy for Fuzzy User Features v. Item-Based, User-Based and Average User Features

solutions proposed in literature is the identification of the link between users' ratings and items' features. This link was defined by the user semantic model that modeled user-features preferences. This model was built by a machine learning algorithm, the Fuzzy C Mean, and used to compute the users' similarities in a user-based CF algorithm. The built of the semantic model and the computing of similarities can be done offline, so, only predictions must be done online. Thus, our approach provides solutions to the *scalability problem*. Therefore, it alleviates the *data sparsity problem* by reducing the dimensionality of data. Users-features matrix has no missing value, thus, similarities between all users can be computed. This has solved the *neighbor transitivity problem*, in which users with similar tastes may not be identified, if they have not both rated any of the same items.

The results obtained, on movies dataset, are encouraging; they improve prediction accuracy compared to standards item-based CF [10] and user-based CF [9], despite the use of one attribute. As futures works, first, we will extend the user semantic model to many attributes. Only the significant attributes must be used, that is to say, those are important to users. Second, we will use the user semantic model to solve the cold start problem in which new items cannot be recommended to users because they haven't any rating. Lastly, we will apply others Data mining algorithms to construct the user semantic model and compare their results.

REFERENCES

[1] M. Balabanovic and Y. Shoham, "Fab: content-based, collaborative recommendation," Comm. ACM, vol. 40, no. 3, pp. 66-72, March 1997, doi:10.1145/245108.245124.

[2] J. C. Bezdek. "Pattern recognition with fuzzy objective function algorithms". Plenum Press, 1981.

[3] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering", Proc of the 14th Conf. on Uncertainty in Artificial Intelligence (UAI '98), July 1998.

[4] R. Burke, "Hybrid web recommender systems", in The Adaptive Web. Lecture Notes in Computer Science, vol. 4321, pp. 377-408, Springer-Verlag, 2007.

[5] M. J. Pazzani, D. Billsus: "Content-based recommendation systems", in The Adaptive Web. Lecture Notes in Computer Science, Vol. 4321, pp. 325-341 Springer-Verlag, 2007.

[6] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating collaborative filtering recommender systems”, ACM Trans. on Information Systems, vol. 22, no. 1, pp. 5–53, 2004.

[7] B. Mobasher, B., X. Jin, and Y. Zhou.: “Semantically enhanced collaborative filtering on the web”, LNAI, vol. 3209, pp. 57-76, Springer Berlin, 2004, Doi: 10.1007/978-3-540-30123-3_4.

[8] MovieLens, http://www.movielens.org/, retrieved: April, 2011.

[9] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl, “Grouplens: an open architecture for collaborative filtering of netnews”, Proc. of Conf. on Computer Supported Cooperative Work, ACM, pp. 175–186, North Carolina, 1994.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” Proc. 10th Int’l WWW Conf., 2001.

[11] X. Su, and T. M. Khoshgoftaar, “A survey of collaborative filtering techniques”. Adv. in Artif. Intell. Jan 2009.

[12] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, “Feature-weighted user model for recommender systems”. Proc of the 11th Int’l Conf. on User Modeling, Corfu, Greece, July 2007.

[13] G. Adomavicius, and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions”, IEEE Trans on Knowledge and Data Engineering, vol.17, no. 6, pp. 734-749, June 2005.

[14] M. Garde, and G. Dudek, “Mixed collaborative and content-based filtering with user-contributed semantic features”. Proc of the 21st National Conf on Art. Intell, vol 2, July, 2006.

[15] P. Melville, R. J. Mooney, and R. Nagarajan, “Content-boosted collaborative filtering for improved recommendations”. In Eighteenth National Conf. on Art. Intell., Canada, July 28 - August 01, 2002.

[16] J. Basilico, and T. Hofmann, “Unifying collaborative and content-based filtering”, Proc. of 21st international conference on Machine learning, pp. 9, July 04-08, 2004, Banff, Alberta, Canada, doi:10.1145/1015330.1015394.

[17] B. Man Kim, Q. Li, C. Seok Park, S. Gwan Kim and J. Yeon Kim, “A new approach for combining content-based and collaborative filters” Journal of Intelligent Information Systems vol. 27, no 1, pp. 79-91, Springer, 2006.

[18] A. Belén Barragáns-Martínez, E. Costa-Montenegro, J. C. Burguillo, M. Rey-López, F. A. Mikic-Fonteand A. Peleteiro “A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition”, In Information Sciences, vol. 15, pp. 4290-4311, Nov. 2010.

[19] C.C. Aggarwal, J.L. Wolf, K-L. Wu, and P.S. Yu, “Horting hatches an egg: A new graph-theoretic approach to collaborative filtering,” Proc. 5th ACM SIGKDD, Int’l Conf. Knowledge Discovery and Data Mining, Aug. 1999.

[20] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” Comm. ACM, vol. 35, no. 12, pp. 61–70, 1992.

[21] M. Ferman, J. Errico., P. Van Breek, and I. Sezan, “Content-based filtering and personalization using structured metadata”. Proc. of the Second ACM/IEEE-CS Joint Conf. on Digital Libraries USA, 2002.

[22] C. Musto, F. Narducci, P. Lops, M. Degemmis, and G. Semeraro, “Content-based personalization services integrating folksonomies”. Proc. E-Commerce and Web Technologies, 10th Int’l Conf., September 2009, Linz, Austria.

[23] M. Pazzani and D. Billsus, “Learning and revising user profiles:The identification of interesting web sites,” Machine Learning, vol. 27, pp. 313-331, 1997.

[24] GroupLens Research Project, www.grouplens.org, retrieved: April, 2011.